Technical Disclosure Commons

Defensive Publications Series

March 2021

FULLY EXTENSIBLE CI/CD PIPELINES

HP INC

Fully Extensible CI/CD Pipelines

Current CI/CD pipeline technologies present limited capacity for extension. Representative of the state-of-the-art are `${{...}}` expressions, which cannot satisfy all users' arbitrary and unexpected requirements for extensibility. The idea disclosed here allows for fully extensible CI/CD pipelines, in a technology-agnostic way. We discuss and exemplify using ADO Pipelines only for the sake of conciseness. Our proposal fits any "pipeline-as-code"-based technology.

As illustrated in *Figure 1*, a manually written pipeline, defined by an original descriptor and all the templates that are directly or indirectly referenced by it are automatically rewritten, as defined by an extensibility descriptor.

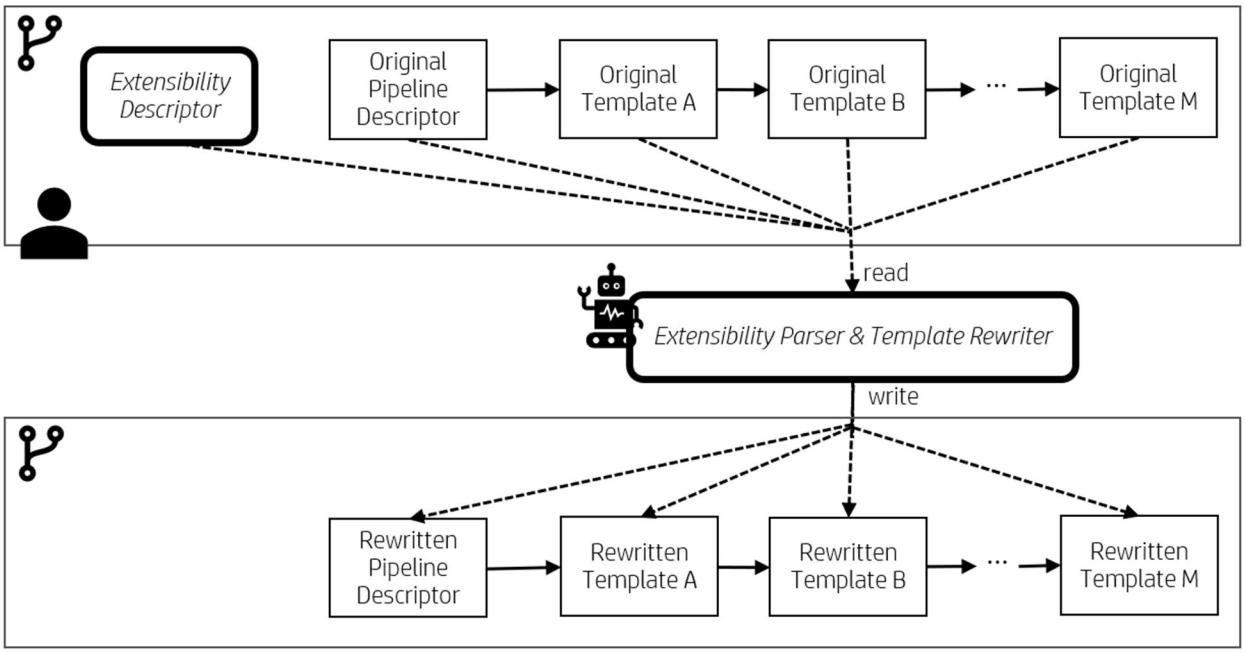

Figure 1: The main components of our proposal.

The exact inner-workings of the `Extensibility Parser & Template Rewriter` component are immaterial for this disclosure. They may be implemented differently, allowing for more or less complex input syntax and file content overwriting. Implementations could be, for instance, based on automatic semi-structured textual transformations (e.g., XSLT). As long as the `Extensibility Parser & Template Rewriter` component is able to automatically parse an extensibility description (whatever its format may be), and effectively rewrite the original files, our proposal should work fine.

If the underlying technology doesn't have built-in support for our proposal, the developer has to mind where the automatically generated are stored. One straightforward way of doing it is to simply store all automatically generated files in a dedicated directory (e.g., `/rewritten`, as used

in the example presented in the next section) under source code management (e.g., Git). If the underlying technology has built-in support for our proposal (ie.,the machines that interpret and run the pipeline commands, aka pipeline agents, have the ability to receive as input not only the pipeline descriptor but also the extensibility descriptor), from a developer perspective, there should be no concern on how to handle and store the automatically generated files.

Illustrative didactic example

Consider the ADO Pipeline defined by the `pipeline-descriptor.yaml` file and the templates presented in the *Code Snippets 1* to *4*. Also, consider the extensibility descriptor presented in *Code Snippet 5*, which is pretty much self-explanatory: it describes a rewriting action that replaces the string `echo "something"` with the string `echo "something ELSE"` in template `/step/step_a.yaml`, which is referenced by template `/job/job_a.yaml`, which is referenced by template `/stages/stage_a.yaml`.

```
stages:
- template: /stages/stage_a.yaml
```

Code Snippet 1: Original file `pipeline-descriptor.yaml`

```
stages:
- stage: Stage_A
  jobs:
  - template: /jobs/job_a.yaml
```

Code Snippet 2: Original file `/stages/stage_a.yaml`

```
jobs:
- job: Job_A
  steps:
  - template: /steps/step_a.yaml
```

Code Snippet 3: Original file `/job/job_a.yaml`

```
steps:
- script: echo "something"
```

Code Snippet 4: Original file `/step/step_a.yaml`

```
[{
  "templatePath": [
           "/stages/stage_a.yaml",
           "/job/job_a.yaml",
           "/step/step_a.yaml"
      ],
      "actions": [{
      "replaceString": {
```

```
            "from": "echo \"something\"",
            "to": "echo \"something ELSE\""
      }
      }]
}]
```

Code Snippet 5: Extensibility Descriptor

A pipeline rewrite could be triggered, for example, by a pre-commit Git hook installed on the Git repository that contains the original pipeline descriptor. The `Extensibility Parser & Template Rewriter` component would then generate *Code Snippets 6* to *9*. Notice that *Code Snippet 9* contains string `echo "something ELSE"` instead of string `echo "something"`, as defined by the extensibility descriptor.

```
stages:
- template: /rewritten/stages/stage_a.yaml
```

Code Snippet 6: Automatically generated file `/rewritten/pipeline-descriptor.yaml`

```
stages:
- stage: Stage_A
  jobs:
  - template: /rewritten/jobs/job_a.yaml
```

Code Snippet 7: Automatically generated file `/rewritten/stages/stage_a.yaml`

```
jobs:
- job: Job_A
  steps:
  - template: /rewritten/steps/step_a.yaml
```

Code Snippet 8: Automatically generated file `/rewritten/job/job_a.yaml`

```
steps:
- script: echo "something ELSE"
```

Code Snippet 9: Automatically generated file `/rewritten/step/step_a.yaml`

Disclosed by Mauricio Moraes, Jhonny Mertz, and Rafael Lopes, HP Inc.